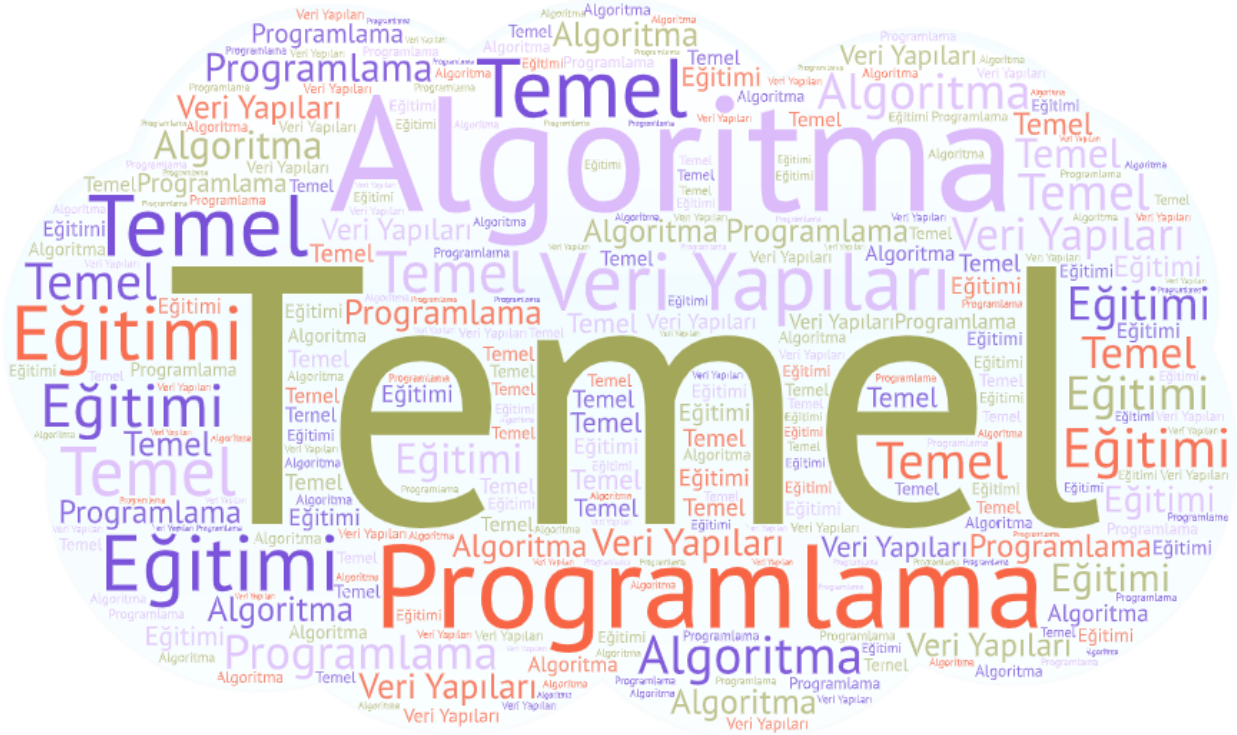
BTK AKADEMİ

ZAFER CÖMERT
Öğretim Üyesi

www.zafercomert.com
comertzafer@gmail.com



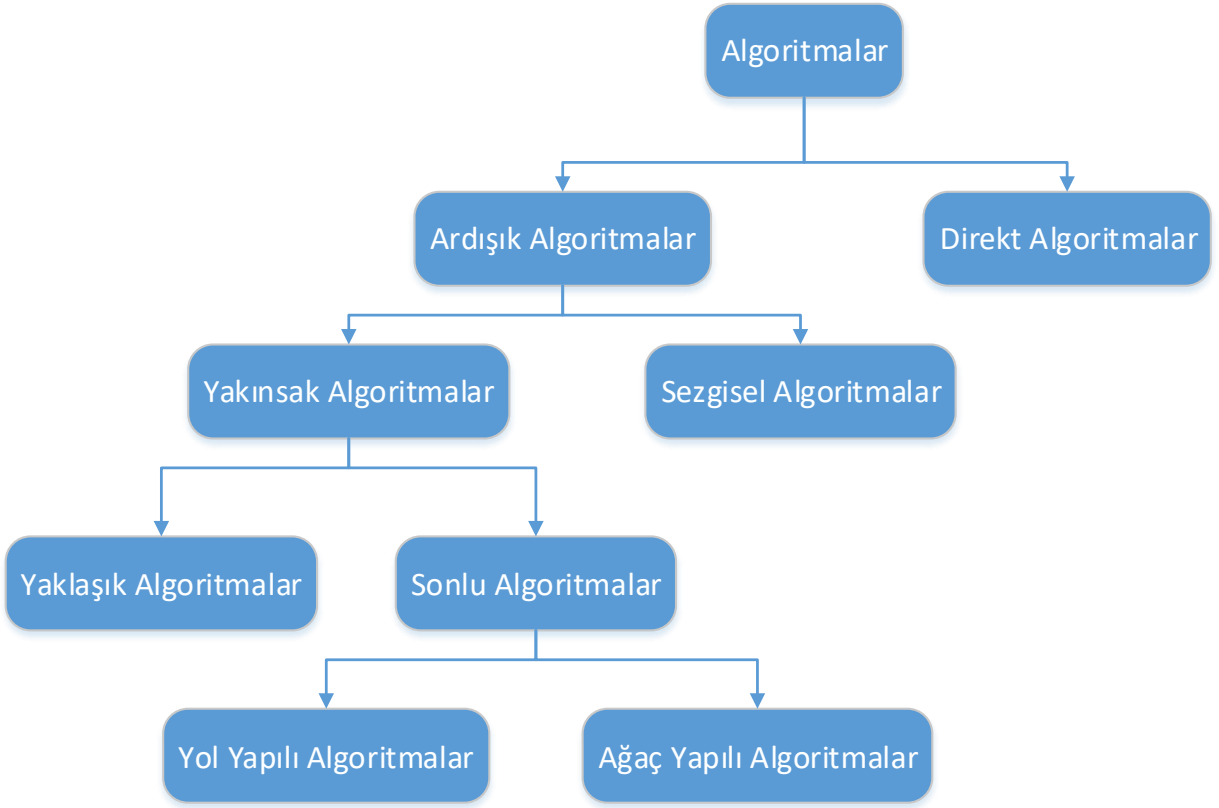
ALGORİTMA TÜRLERİ

Algoritma Türleri

- Algoritmalar prosedürleri işletme şekillerinden dolayı farklı kategorilere ayrılmaktadır.

- Bu farklılıklar algoritmaların çalışma şekillerine ve yapılarına yansımaktadır.

Algoritma Türleri



Direkt Algoritmalar

- Algoritmaların en temel çalışma bileşenlerinden biri iterasyonlardır.

- İterasyon, belirli koşullar altında kendini tekrar eden kod blokları olarak ifade edilebilir.

- İterasyonalarla **çalışmayan** algoritmalar direkt algoritmalar olarak isimlendirilir.

Direkt Algoritmalar

$$ax^2 + bx + c = 0$$

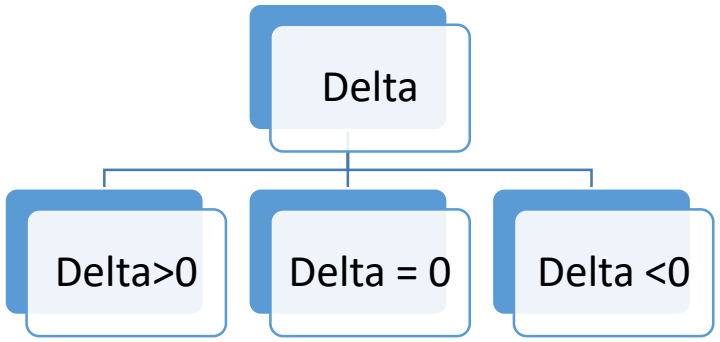
- Yukarıda ikinci dereceden bir denklemin tanımı yapılmıştır.

- Söz konusu denklemin çözümü direkt algoritmalar ile doğrudan gerçekleştirilebilir.

Direkt Algoritmalar

$$ax^2 + bx + c = 0$$

$$\Delta = b^2 - 4ac$$



$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

$$x_1 = \frac{-b}{2a}$$

$$x_2 = x1$$

Denklemin reel kökü yoktur.

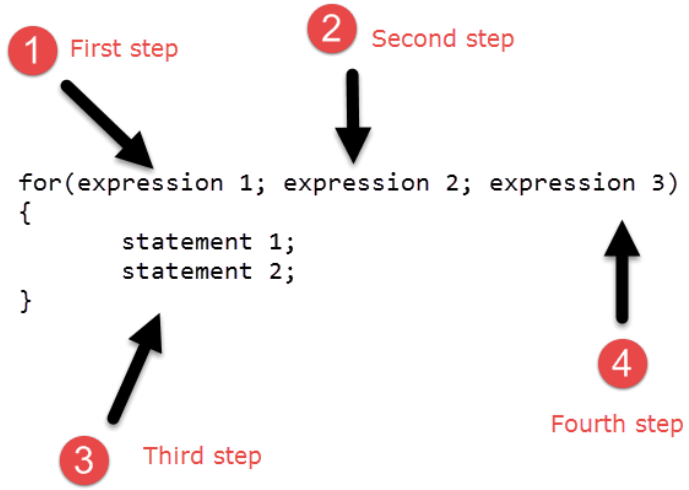
Direkt Algoritmalar

- Yukarıdaki ifadeler doğrultusunda ilgili problemi çözmek için herhangi bir iteratif yapıya ihtiyaç olmadığı ve kodlamanın doğrudan yapılabileceği görülmektedir.

- Bu şekilde iterasyon ile **çalışmayan** algoritmalar direkt algoritmalar olarak ifade edilir.

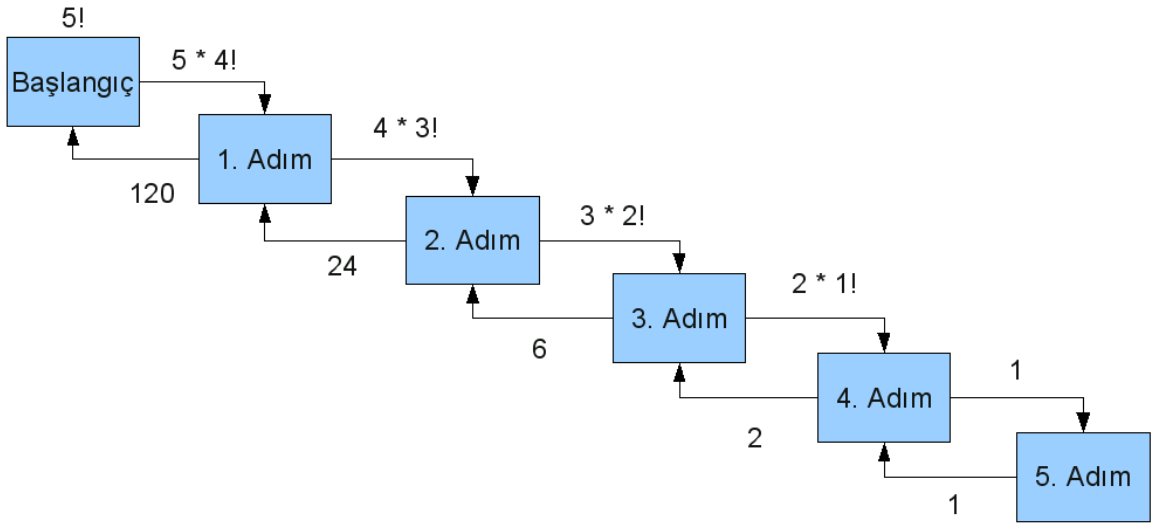
Ardışık Algoritmalar

- Direkt algoritmaların aksine amaçlanan problemin çözümünü yerine getirmek üzere pek çok algoritma ardışık olarak çalışır ve bu algoritmalar ardışık algoritmalar olarak adlandırılır.



Ardışık Algoritmalar

- Bir örnek ile devam etmek üzere faktöriyel hesabına bakalım:



Ardışık Algoritmalar

- Bir örnek ile devam etmek üzere faktöriyel hesabına bakalım:

$$n! = \prod_{k=1}^{n} k = 1 * 2 * \cdots * (n-1) * n$$

$$n! = \begin{cases} 1 & n \leq 1 \\ n.(n-1)! & n > 1 \end{cases}$$

Ardışık Algoritmalar

- Bir örnek ile devam etmek üzere faktöriyel hesabına bakalım:

```
static void FaktoriyelHesabi()
{
    int n = Convert.ToInt32(
        Console.ReadLine());
    int f = 1;
    for (int i = 2; i <= n; i++)
        f *= i;
    Console.WriteLine("{0}! = {1}", n, f);
}
```

Ardışık Algoritmalar

- Bir örnek ile devam etmek üzere faktöriyel hesabına bakalım:

```
static int FaktoriyelHesabi2(int n)
{
    if (n <= 1)
        return 1;
    return n * FaktoriyelHesabi2(n - 1);
}
```

Yakınsak Algoritmalar

- Aranılan çözüme doğru yakınsayan ardışık algoritmalardır.
- Bazı yakınsak algoritmalar kesin çözümü elde edemezler, fakat bu çözüme yaklaşık bir değeri kesin çözüm alırlar.
- Yaklaşık algoritmalar sonlu değillerdir; fakat her bir ileri iterasyon onları kesin çözüme biraz daha yaklaştırır.
- Yaklaşık algoritmalara Değişken Kesen Metodu, Arama Teknikleri vb. çok bilinen birkaç örnek verilebilir.

Sonlu Algoritmalar

- Sonlu algoritmalar, iterasyonların sonlu bir sayısında kesin çözümü garanti eden yakınsak algoritmalardır ve kendi arasında yol yapılı ve ağaç yapılı olmak üzere ikiye ayrılırlar.

Sonlu Algoritmalar

Yol Yapılı

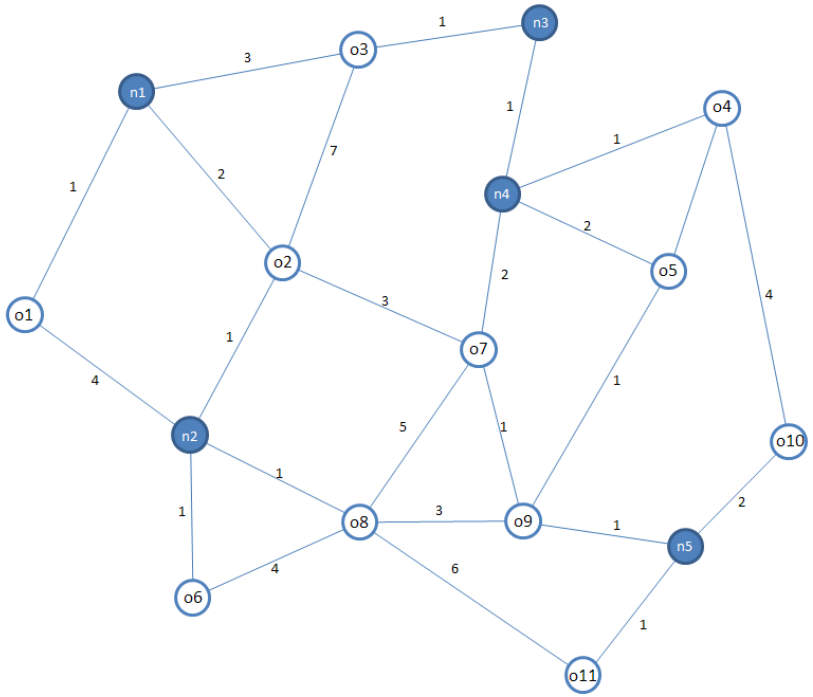
- Sonlu algoritmaların pek çoğu yol yapısına sahiptir.
- Bu yol yapısında bir iterasyon bir önceki iterasyonu iterasyon dizilerinde farklı dallar üretmeksizin takip eder.

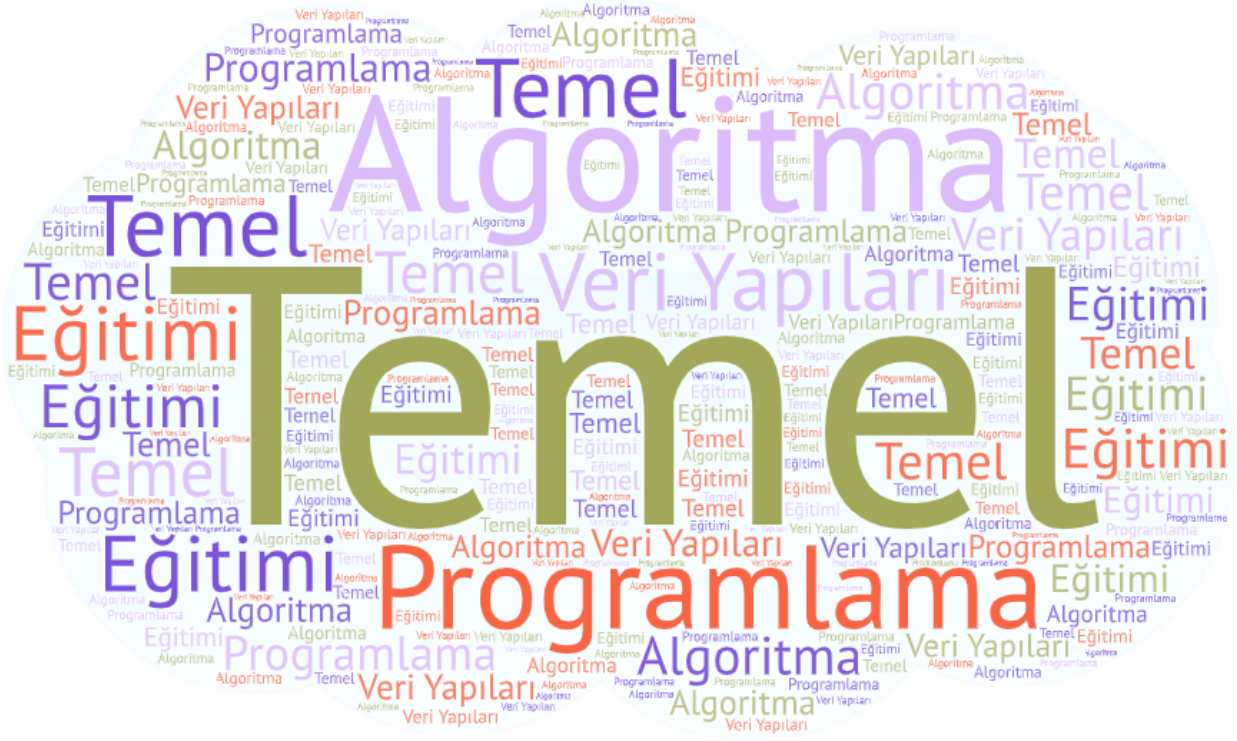
Ağaç Yapılı

- Diğer sonlu algoritmalarda iterasyon dizileri, pek çok paralel dalları içeren bir ağaç şeklindedir.
- Birçok ağaç arama algoritmaları bu sınıfa aittir.

Gezgin Satıcı Problemi

- Seyyar satıcı problemi şu şekilde tanımlanabilir:
- Bir seyyar satıcı var;
- Bu satıcı, mallarını n şehirde satmak istiyor;
- Öte yandan, mantıklı bir şekilde, bu satıcı bu şehirleri mümkün olan en kısa şekilde ve her bir şehre maksimum bir kere uğrayarak turlamak istiyor.
- Problemin amacı, satıcıya bu en kısa yolu sunabilmektir.





ALGORİTMA TÜRLERİ

www.zafercomert.com
comertzafer@gmail.com